design ideas

Edited by Bill Travis and Anne Watson Swager

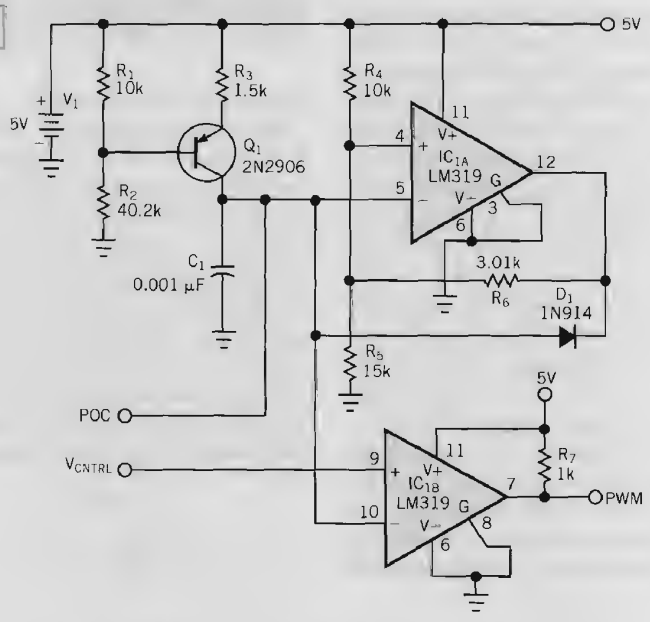
Simple circuit provides efficient PWM

Kenneth Levine, Eldec, Lynwood, WA

You often need to control the duty cycle of a square wave. The circuit in **Figure 1** works from a single 5V supply at a frequency of 100 kHz. With minor component changes, you can configure it for a range of frequencies. The circuit consists of a ramp (sawtooth) oscillator and a comparator. The circuit compares the ramp to an externally applied voltage, V_{CNTRL}. When V_{CNTRL} is greater than the ramp voltage, the PWM output is 5V. The sawtooth oscillator consists of a current source (R_1, R_2, R_3, and Q_1), a timing capacitor (C_1), and the oscillator-control circuitry (R_4, R_5, R_6, and IC_{1A}). The oscillator-control circuitry sets the upper and lower voltages of the timing capacitor at 3 and 1V.

R_1 and R_2 set the voltage at Q_1's base to 1V. R_3 and the V_{BE} drop (0.7V) are in parallel with R_1, so the voltage across R_3 is 0.3V for a current of 200 mA. Because the emitter current greatly exceeds the base current, the collector current is nearly 200 mA. A capacitor receiving its charge from a current source charges linearly. The current in C_1 is 200 mA, and the change in voltage is 2V (1 to 3V). From I=CdV/dt, dt=10 msec, and the frequency is 100 kHz. The oscillator-control circuitry uses a comparator with hysteresis to compare the capacitor (ramp) voltage with the charge limits. An externally applied power-on-clear (POC) signal ensures reliable start-up. POC must connect to an open-collector device that pulls to ground at power-on and produces no voltages when inactive.

Figure 1



A dual comparator and a handful of components configure an inexpensive PWM generator.

After removal of POC, IC_{1A}'s Pin 5 is at ground, IC_{1A}'s Pin 12 is at open-collector status, and IC_{1A}'s Pin 4 is at 3V (the voltage across R_5). C_1 now starts charging. When the voltage on C_1 exceeds 3V, IC_{1A}'s Pin 12 switches to nearly ground, IC_{1A}'s Pin 4 drops to 1V (the voltage across the parallel combination of R_5 and R_6), and the charge on C_1 rapidly bleeds off through D_1. The discharge time is short enough to have a negligible effect on the oscillation frequency. When the voltage on C_1 drops below 1V, IC_{1A}'s Pin 12 switches to open-collector status, D_1 stops conducting, and C_1 begins to charge again. The entire cycle repeats, creating sawtooth oscillation.

The circuit creates PWM by comparing the voltage on C_1 with V_{CNTRL}. As long as V_{CNTRL} is greater than the voltage on C_1, the PWM output stays at 5V. You should limit V_{CNTRL} to a bit more than 1V and a bit less than 3V to prevent comparator oscillation. If the source of V_{CNTRL} can tolerate it, you can add hysteresis. The relationship of PWM to V_{CNTRL} is linear: PWM width=50(V_{CNTRL}21)%, valid for 1V<V_{CNTRL}<3V. To increase the operating frequency, you can replace the LM319 with a faster comparator. (DI #2309).

To Vote For This Design,
Circle No. 385

Visual Basic models MDAC offset

Olga Belousava, Los Alamos, NM, and Alex Belousov, New York, NY

It's hard to imagine that, for such an old-hat item as a standard R-2R multiplying DAC (MDAC), there still exists some "dark area" in modeling and calculating its dc offset, V_{OFF}, and related output resistance, R_O. You can obtain some information from **references 1** and **2** and other references regarding the code dependency of R_O and V_O, but the simplified formulas given therein are insufficient for thorough engineering analysis/design and computer modeling/simulation. Moreover, these formulas apply mainly to the case in which the Ref erence pin (**Figure 1**) is open, whereas most MDAC applications connect the Reference pin to a low-impedance source. Also, the source resistance, R_{IN}, has an impact on V_{OFF} and R_O. This Design Idea introduces an equivalent circuit for the MDAC and discusses mathematical models. A software program, "DAC Designer AO," simplifies the offset calculations. You can download the Visual Basic files from *EDN*'s Web site, www.ednmag.com. At the registered-user area, go into the Software Center to download the files from DI-SIG, #2305.

TABLE 1–MAXIMUM OFFSET WITH REF PIN GROUNDED

N (bits)	Decimal	Hex	Binary (maximum)	V_{OFF}
8	235	00EB	11101011	$2.652*V_{OS}$
9	469	01D5	111010101	$2.763*V_{OS}$
10	939	03AB	1110101011	$2.875*V_{OS}$
11	1877	0755	11101010101	$2.986*V_{OS}$
12	3755	0EAB	111010101011	$3.097*V_{OS}$

The models are based on the idealized R-2R resistive ladder in **Figure 1** with zero on-resistance and infinite off-resistance in the switches and without current leakage or parasitic voltages. **Figure 2** gives an equivalent reciprocal-π network for the ladder, in which G_{11} is the input conductance with the output terminals shorted, G_{12} is the transfer conductance, and G_{22} is the output conductance with the input terminals shorted. The applicable math formulas are as follows:

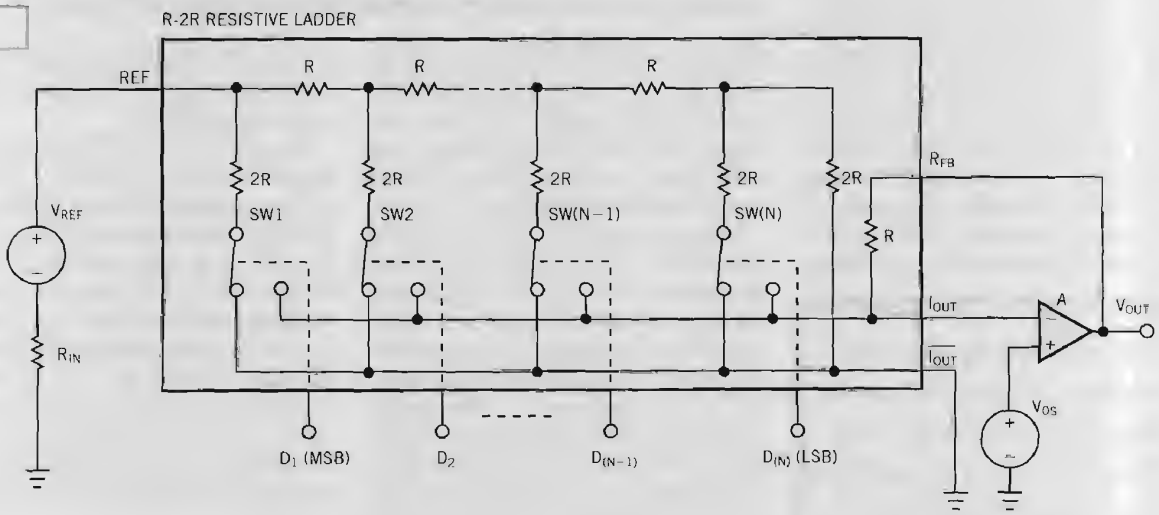
$$G_{11} = 1/R,$$

$$G_{12} = \sum_{i=1}^{N}(D_i - 2^{-i})/R,$$

$$G_{22} = \left(\sum_{i=1}^{N}(D_i(1+2\bullet 4^{-i}) - \sum_{i=2}^{N-1}(D_i(1-4^{-i})\bullet \sum_{j=i+1}^{N}(D_j\bullet 2^{-j}))\right)/3R.$$

It's obvious that G_{11} has a constant value, the reciprocal of the base resistance, R. G_{12} is a linear function of the input code and base resistance R. The expression for G_{22}, however, reveals complex, nonlinear behavior as a function of the digital input code. A Thevenin transform in **Figure 2**'s circuit produces the simplified equivalent circuit in **Figure 3**. The output resistance, R_O, in the most common case, when the MDAC connects to

Figure 1



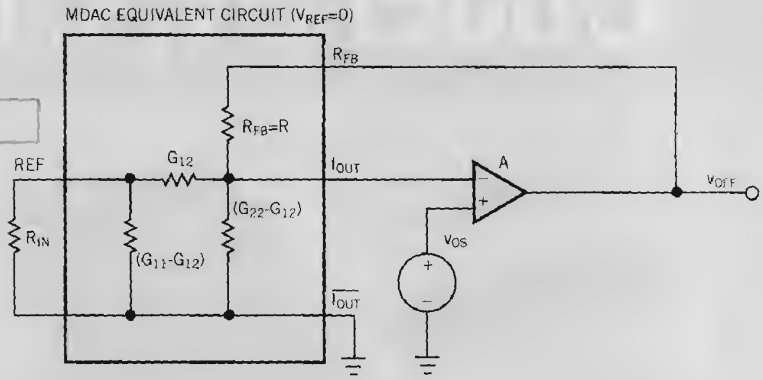
The output impedance of a multiplying R-2R DAC is a complex, nonlinear function (third equation in the text) of the digital input code.

a low-resistance source, is the reciprocal of G_{22}. The offset voltage is thus

$$V_{OFF} = V_{OS} \bullet (1 + R \bullet G_{22}).$$

Table 1 gives the maximum theoretical values for V_{OFF} in 8- to 12-bit MDACs. You can easily obtain the exact values for V_{OFF} and R_O for any arbitrarily chosen input code by using the cited software program. The program also allows you to compute the statistics (maximum V_{OFF}, its mean, and standard deviation) for any predefined range of input codes. You can use the suggested equivalent circuit and mathematical models with any computer simulation packages in dc-analysis mode. (DI #2305).

Figure 2

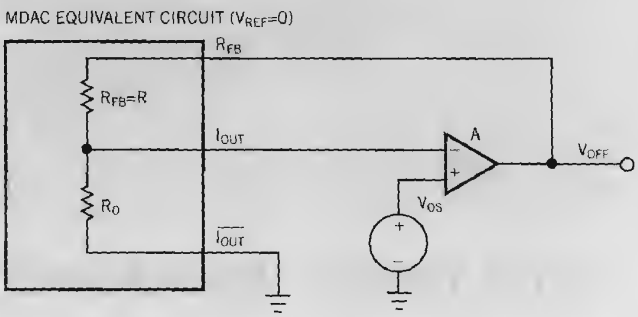


The input conductance, transconductance, and output conductance of the MDAC in Figure 1 provide a convenient simplification of the circuit.

REFERENCES

1. Sheingold, Daniel H, Editor, *Analog-Digital Conversion Databook*, Prentice-Hall, 1986.
2. *Data Converter Reference Manual*, Volume 1, pg 2-540 to 2-541, Analog Devices, 1992.

Figure 3



A Thevenin transformation of the equivalent circuit in Figure 2 produces a greatly simplified circuit.

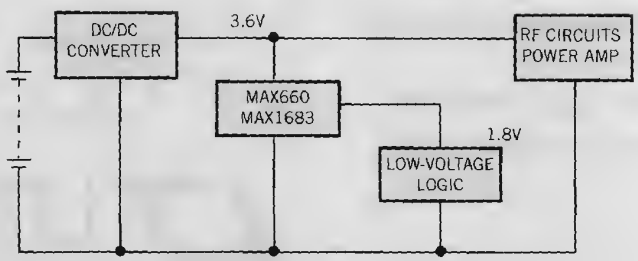
To Vote For This Design, Circle No. 386

Charge-pump circuit divides by two

Budge Ing, Maxim Integrated Products, Sunnyvale, CA

Small size and efficiency approaching 100% make switched-capacitor charge pumps popular for voltage doubling and inverting in miniature dc-dc applications. Few are aware, however, that most charge pumps can halve as well as double or invert an input voltage. The increasing adoption of low-voltage logic makes this ÷ 2 capability useful for generating low-voltage supplies in portable equipment. You can use it, for example, to convert a 3.6V RF-supply voltage to 1.8V for powering low-

Figure 1

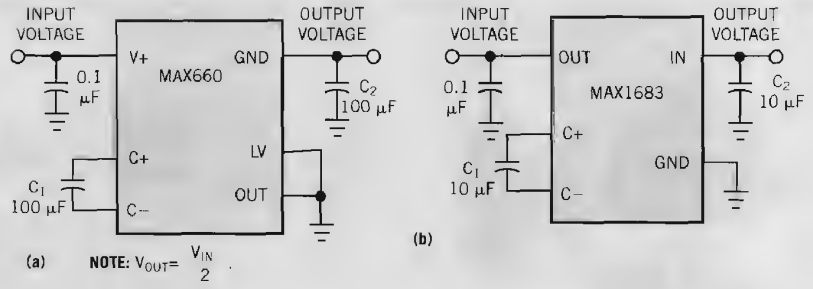


Simply reversing a voltage-doubler charge pump converts it to a voltage divider.

voltage logic (**Figure 1**). Simply reversing the input and output of a voltage doubler makes it a voltage divider. Implementing this scheme with MAX660 or MAX1683 voltage-doubler charge pumps requires only three external capacitors (**Figure 2**). Both configurations in **Figure 2** accept input voltages of 3.6 to 10V, but they present trade-offs in size and output-current capability.

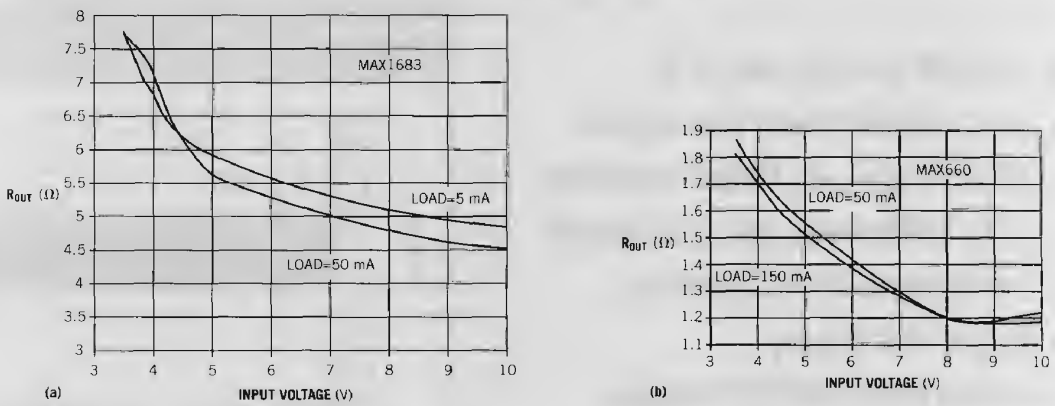
For 3.6V inputs, the robust MAX660, which comes in an eight-pin DIP or SO package, delivers 150 mA with efficiency greater than 88% and an output-voltage drop of less than 300 mV. If you require a smaller package, the MAX1683 (avail-

Figure 2



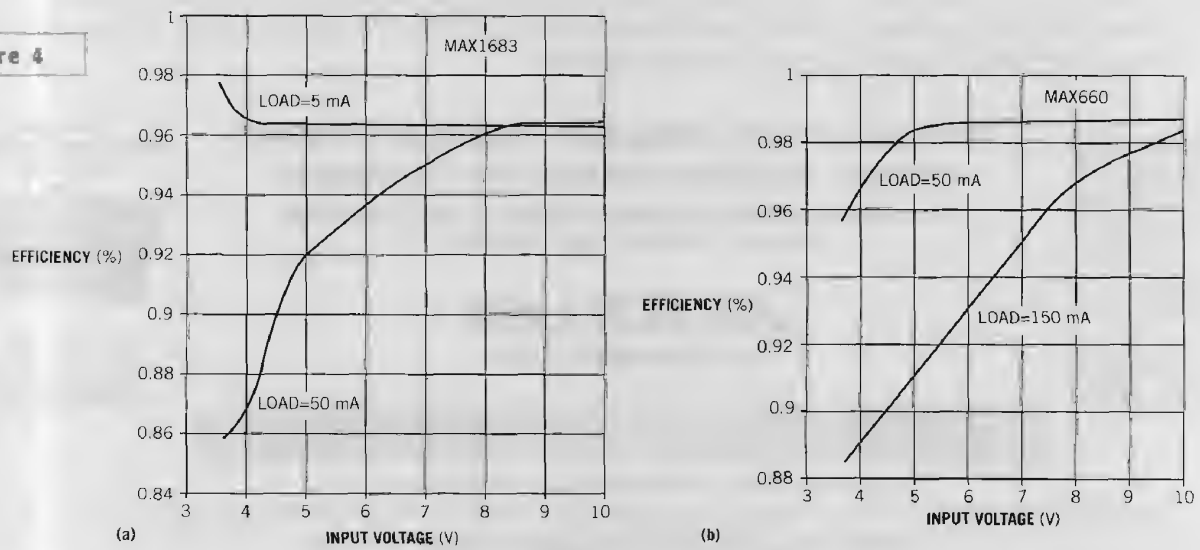
A MAX660 voltage doubler makes an efficient 150-mA voltage divider (a); the smaller MAX1683 does the same for applications requiring only 50 mA (b).

Figure 3



Over the usable 3.6 to 10V input-voltage range, the small MAX1683 (a) exhibits higher output resistance than the more robust MAX660 (b).

Figure 4



Both the MAX1683 (a) and the MAX660 (b) offer better than 90% efficiency over a large portion of their usable operating range.

able in a five-pin SOT-23) offers 50-mA capability with a 3.6V input and as much as 100 mA with inputs higher than 8V. Its efficiency is 97% at 5 mA and 86% at 50 mA. Each device has an internal clock. The MAX660 runs at a nominal 10 kHz (with the FC pin open), and the MAX1683 runs at a nominal 35 kHz. Each divider's output resistance depends on the internal clock frequency, the flying capacitor (C_1), the resistance of the internal switches, and the resistance of the output capacitor C_2. You can calculate the output resistance by:

$$R_{OUT} = \frac{1}{f_{OSC}C_1} + 4(R_4 + R_2 - R_1 - R_3) + 2R_{ESR},$$

where f_{OSC} is the oscillator frequency, R_1 to R_4 are the $R_{DS(ON)}$ values for the four internal switches, and R_{ESR} is the equivalent series resistance for the output capacitor, C_2. The graphs in **Figure 3** illustrate the performance of the charge pumps operating in voltage-divider mode. **Figure 3a** shows the output resistance versus input voltage for the MAX1683. **Figure 3b** shows the same parameters for the MAX660. **Figure 4** shows efficiency versus input voltage for the MAX1683 (**Figure 4a**) and the MAX660 (**Figure 4b**). (DI #2301).

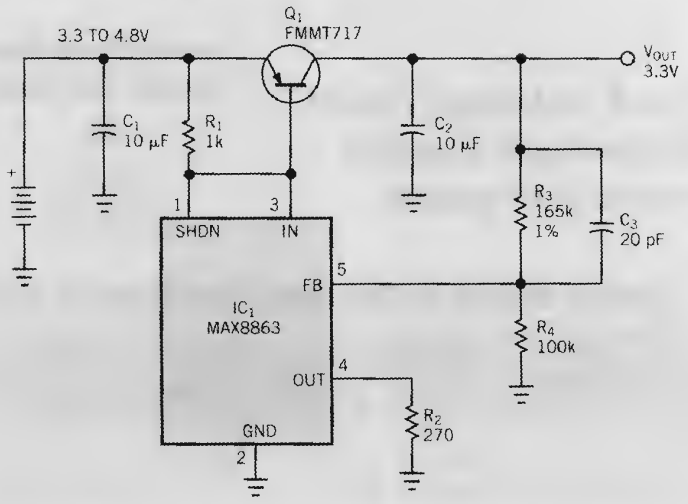
To Vote For This Design,
Circle No. 387

Pass transistor lowers dropout voltage

Matt Schindler, Maxim Integrated Products, Sunnyvale, CA

With linear regulators, you measure dropout voltage, $V_{IN} - V_{OUT}$, at the minimum input voltage for which the IC sustains regulation. Low dropout means longer battery life, because the load circuit continues to operate while the battery discharges to a lower terminal voltage. The external transistor helps to form a linear-regulator circuit whose dropout voltage at 100-mA load current is only 10 mV (**Figure 1**). (The linear-regulator IC by itself specs a 100-mV dropout at 100 mA.) The external transistor also boosts the maximum available load current to 1A.

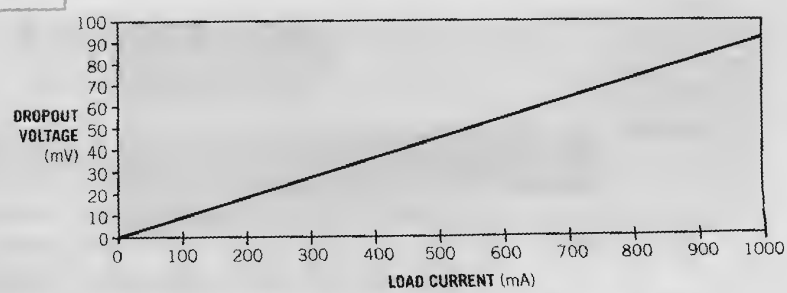
Figure 1



Unorthodox transistor connections to a low-dropout regulator allow you to squeeze a 100-mV dropout voltage down to 10 mV.

Unorthodox connections enable the IC to drive Q_1. Connecting Pin 3 to the transistor's base allows base current to flow through the internal switching MOSFET, out of Pin 4, and through R_2 to ground. The MOSFET then regulates V_{OUT} by controlling Q_1's base current. Because C_2 sets a dominant pole that stabilizes the loop, you should choose a ceramic type or other low-ESR capacitor. C_2 improves the phase margin by forming a pole-zero combination that increases the phase at crossover. Q_1 saturates when the battery voltage drops low enough for V_{OUT} to drop out of regulation, and R_2 limits the base current for that condition to approximately 10 mA. Q_1's collector-emitter voltage at saturation, 10 mV with 1-mA base current and 100-mA collector current, sets the dropout voltage for these conditions.

Figure 2



Dropout voltage for the circuit in Figure 1 varies from 10 mV at 100 mA to 90 mV at 1A.

The measured dropout voltage varies with load current (**Figure 2**). The circuit delivers as much as 1A at 3.3V. You can adjust the output from 5.5V down to 1.25V using the formula $V_{OUT}=1.25\{1+(R_3/R_4)\}$, with appropriate changes to the value of R_2, using the formula $R_2=(V_{IN(MIN)}-0.7V)/10$ mA. Small components allow the entire circuit to occupy less than 0.24 in.^{2} of board area. (IC_1 comes in an SOT-23 package.) (DI #2323).

PIC μC implements CRC-16 algorithm

Lon Glastner, Solutions Cubed, Chilo, CA

Detecting errors in serial data can be paramount in completing an embedded-control design. Determining which algorithm to use for detecting serial-communications errors depends on several factors. Ideally, the method should require minimal hardware and little computational power from your processor and still provide high-level protection against undetected data errors. The cyclic redundancy check (CRC) combines all these factors under one umbrella. A multitude of CRC flavors exists, including the Dow CRC (8 bits), CRC-16, and CRC-CCITT (both 16 bits). The CRC-16 uses a 16-bit shift register and can detect the following error types:

- any cluster of errors within a 16-bit section of data,
- any odd number of errors within the data field,
- all double-bit errors in the data field, and
- most large clusters of errors.

TABLE 1–XOR TRUTH TABLE

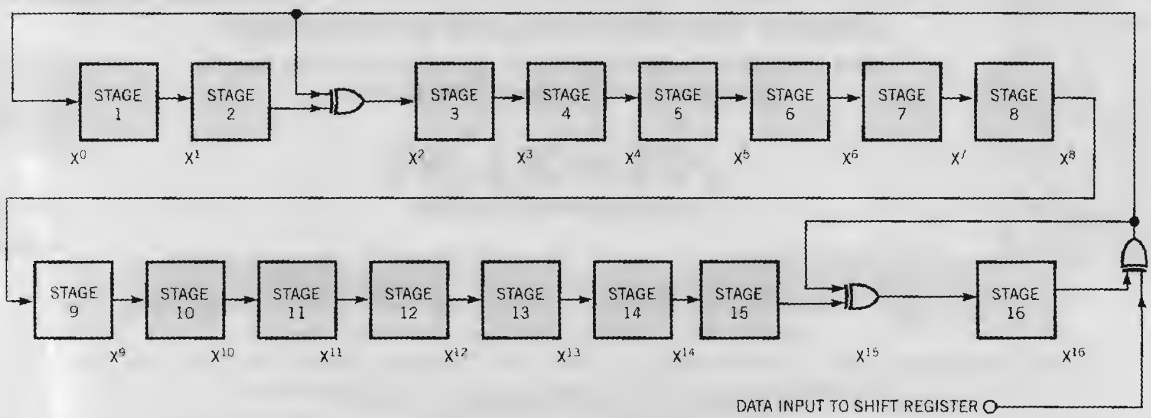
X	Y	XOR
0	0	0
0	1	1
1	0	1
1	1	0

You can characterize the CRC-16 as both a polynomial expression and as a hardware-based shift register (**Figure 1**). You can implement a CRC-16 in a midrange PIC μC with minimal coding and without additional hardware. Selecting a μC with an on-chip USART, such as the PIC16C63, eases serial communications. This implementation does not focus on the mathematical proof of a CRC-16's error-correcting effectiveness. The CRC algorithm is so effective it's an industry-accepted method for detecting data errors. The heart of a CRC-16 algorithm is a shift register. You generate the shift register by shifting each data bit through the algorithm. In this implementation, the data shifts the most significant bit first, one data byte at a time.

Two temporary registers buffer the data to prevent the shifting from corrupting the data. The shift register comprises two separate 8-bit registers, CRC16_HI and CRC16_LO. The most significant bit of CRC16_HI is the location of Stage 16 (**Figure 1**). From the **figure** and **Table 1**, you can see that the result of XORing the input data bit and the contents of Stage 16 of the shift register determines the effect that new data has on the shift register. If the result is a one, then you must complement the contents of stages 2 and 15 before rotating the new data into the shift register. If the result is a zero, then the new data can rotate immediately into the shift register. Some housekeeping tips can be helpful here. The data transmitter should generate its CRC-16 in the same manner as the data receiver. Also, it's advisable to clean the CRC-16 shift register before rotating the first data bit of the data string into it.

Figure 1

(a) POLYNOMIAL REPRESENTATION OF CRC-16: $X^{16}+X^{15}+X^2+1$
(b) HARDWARE REPRESENTATION OF CRC-16



You can represent the CRC-16 algorithm as both a polynomial (a) and a hardware-based shift register (b).

How does the CRC-16 identify data errors? The simplest method is to attach the shift register to the end of the data string. In this implementation, the CRC16_HI register should follow the last byte of data sent. The CRC16_LO register follows the CRC16_HI register. If the receiving system computes a CRC-16 value from all the data bytes and the attached shift register (CRC16_HI and CRC16_LO), then the resulting CRC-16 code is 0000h. Any nonzero result indicates an error in the data. In some systems, an error may occur that results in all zeros being sent as the data and attached CRC-16. This type of error poses as error-free data. In these systems, you can overcome the false indication by complementing the CRC-16 before attaching it to the data string. The CRC-16 shift register generated by attaching the complement is always 800Dh.

The code fragment in **Listing 1** generates a CRC-16 shift register on a byte-by-byte basis. You could embed this code within serial-receive and serial-send routines to provide a powerful error-detection tool. You can easily generate the CRC-16 on the fly, thus minimizing the use of processor resources. You can download the listing from *EDN*'s Web site, www.ednmag.com. At the registered-user area, go into the Software Center to download the files from DI-SIG, #2321. (DI #2321).

To Vote For This Design, Circle No. 389

LISTING 1—PIC CODING FOR CRC-16 SHIFT REGISTER

```
;*****************************************************************************
;MAKE_CRC16 - This routine generates a CRC-16 shift register from the data byte stored
;in the DATA_REG register.  DATA_TEMP0 and DATA_TEMP1 are used to buffer the data
;and prevent it from being corrupted by the CRC-16 generation process.
;*****************************************************************************
MAKE_CRC16
        movf    DATA_REG,W
        movwf   DATA_TEMP0              ;Store data in temporary register
        movlw   '00001000'b             ;Set counter for 8 data bits
        movwf   NUMBER_BITS             ;Load counter register
More_Rotates
        movf    DATA_TEMP0,W            ;Move buffered data to 2nd buffer
        movwf   DATA_TEMP1              ;This register is corrupted with every pass
        movf    CRC16_HI,W              ;Move upper shift register to working reg.
        xorwf   DATA_TEMP1              ;XOR shift register with data register
        btfss   DATA_TEMP1,7            ;MSB is XOR of stage16 and input data bit
        goto    No_Xorwf                ;If bit is clear then no complement of stage2,15
        movlw   '00000010'b             ;Prepare to complement stage2
        xorwf   CRC16_LO                ;Complement stage2 of shift register
        movlw   '01000000'b             ;Prepare to complement stage15
        xorwf   CRC16_HI                ;Complement stage15 of shift register
No_Xorwf
        rlf     DATA_TEMP0              ;Rotate next data bit into position
        rlf     DATA_TEMP1              ;Rotate XOR of input into CRC16_LO
        rlf     CRC16_LO                ;Shift CRC16 register
        rlf     CRC16_HI                ;Shift CRC16 register
        decfsz  NUMBER_BITS             ;Count out 8 data bits
        goto    More_Rotates            ;Not finished with this data byte
        return                          ;This byte is finished
```

Monostable makes low-cost F/V converter

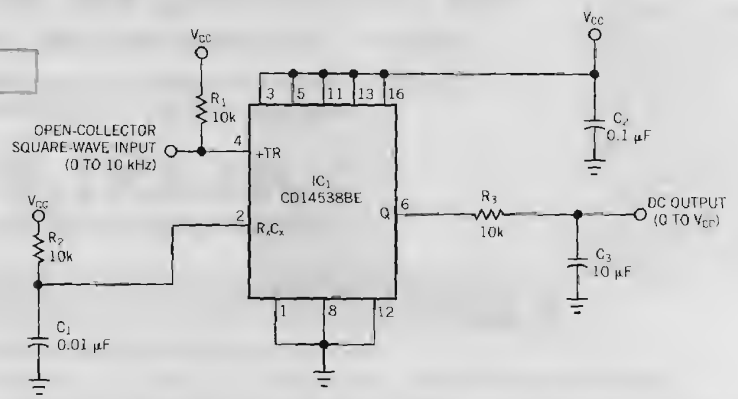
Mark Brinegar, Dart Controls Inc, Zionsville, IN

The circuit in **Figure 1** is a low-cost frequency-to-voltage (F/V) converter. Using a monostable (one-shot) multivibrator, the circuit accepts an open-collector square wave that varies in frequency from 0 to 10 kHz. The one-shot produces a pulse of a fixed width each time the input signal triggers it. The result is a variable-frequency, variable-duty-cycle signal at the output of the one-shot. The time constant determined by R_2C_1, 100 µsec, determines the width of the pulses the one-shot produces. This time matches the period of the maximum input frequency (10 kHz). The duty cycle of the one-shot's output is thus 100% when the input is at its maximum frequency.

The variable-frequency, variable-duty-cycle output of the one-shot is the input for the lowpass filter comprising R_3 and C_3. The net result is, as the input frequency varies from 0 to 10 kHz, the dc output signal varies from 0V to V_{CC}. You can alter the circuit to accommodate different input frequencies by simply adjusting the R_2C_1 time constant to match the period of the desired maximum input frequency. (DI #2322).

To Vote For This Design, Circle No. 390

Figure 1



A single CMOS one-shot multivibrator provides a simple and inexpensive frequency-to-voltage conversion function.